**Mongodb学习二**

# 操作符

## 条件操作符

$eq Matches values that are equal to a specified value.

$gt Matches values that are greater than a specified value.

$gte Matches values that are greater than or equal to a specified value.

$in Matches any of the values specified in an array.

$lt Matches values that are less than a specified value.

$lte Matches values that are less than or equal to a specified value.

$ne Matches all values that are not equal to a specified value.

$nin Matches none of the values specified in an array.

db.inventory.find( { qty: { $eq: 20 } } )

db.inventory.find( { qty: { $in: [ 5, 15 ] } } )

db.inventory.find( { qty: { $nin: [ 5, 15 ] } } )

## 逻辑操作符

| **Name** | **Description** |
| --- | --- |
| [$and](https://docs.mongodb.com/manual/reference/operator/query/and/#op._S_and) | Joins query clauses with a logical AND returns all documents that match the conditions of both clauses. |
| [$not](https://docs.mongodb.com/manual/reference/operator/query/not/#op._S_not) | Inverts the effect of a query expression and returns documents that do *not* match the query expression. |
| [$nor](https://docs.mongodb.com/manual/reference/operator/query/nor/#op._S_nor) | Joins query clauses with a logical NOR returns all documents that fail to match both clauses. |
| [$or](https://docs.mongodb.com/manual/reference/operator/query/or/#op._S_or) | Joins query clauses with a logical OR returns all documents that match the conditions of either clause. |

{ $and: [ { <expression1> }, { <expression2> } , ... , { <expressionN> } ] }

db.inventory.find( { $and: [ { qty: { $gt: 12 } }, { qty: { $lt: 40 } } ] } )

{ $not: [ { <expression1> }, { <expression2> }, ... { <expressionN> } ] }

db.inventory.find( { price: { $not: { $gt: 1.99 } } } )

{ $nor: [ { <expression1> }, { <expression2> }, ... { <expressionN> } ] }

db.inventory.find( { $nor: [ { price: 1.99 }, { sale: true } ] } )

{ $or: [ { <expression1> }, { <expression2> }, ... , { <expressionN> } ] }

db.inventory.find( { $or: [ { quantity: { $lt: 20 } }, { price: 10 } ] } )

允许嵌套

db.inventory.find({$or:[{$and:[{qty:{$gt:12}},{qty:{$lt:40}}]}, {qty:10}]})

## 元素操作符

### $type

| **Type** | **Number** | **Alias** | **Notes** |
| --- | --- | --- | --- |
| Double | 1 | “double” |  |
| String | 2 | “string” |  |
| Object | 3 | “object” |  |
| Array | 4 | “array” |  |
| Binary data | 5 | “binData” |  |
| Undefined | 6 | “undefined” | Deprecated. |
| ObjectId | 7 | “objectId” |  |
| Boolean | 8 | “bool” |  |
| Date | 9 | “date” |  |
| Null | 10 | “null” |  |
| Regular Expression | 11 | “regex” |  |
| DBPointer | 12 | “dbPointer” | Deprecated. |
| JavaScript | 13 | “javascript” |  |
| Symbol | 14 | “symbol” | Deprecated. |
| JavaScript (with scope) | 15 | “javascriptWithScope” |  |
| 32-bit integer | 16 | “int” |  |
| Timestamp | 17 | “timestamp” |  |
| 64-bit integer | 18 | “long” |  |
| Decimal128 | 19 | “decimal” | New in version 3.4. |
| Min key | -1 | “minKey” |  |
| Max key | 127 | “maxKey” |  |

{ field: { $type: <BSON type> } }

{ field: { $type: [ <BSON type1> , <BSON type2>, ... ] } }

db.addressBook.insertMany(

[

{ "\_id" : 1, address : "2030 Martian Way", zipCode : "90698345" },

{ "\_id" : 2, address: "156 Lunar Place", zipCode : 43339374 },

{ "\_id" : 3, address : "2324 Pluto Place", zipCode: NumberLong(3921412) },

{ "\_id" : 4, address : "55 Saturn Ring" , zipCode : NumberInt(88602117) }

]

)

db.addressBook.find( { "zipCode" : { $type : 2 } } );

db.addressBook.find( { "zipCode" : { $type : "string" } } );

### $exists

表示field存在或者不存在

{ field: { $exists: <boolean> } }

db.inventory.find( { qty: { $exists: true, $nin: [ 5, 15 ] } } )

## 评估操作符

| **Name** | **Description** |
| --- | --- |
| [$expr](https://docs.mongodb.com/manual/reference/operator/query/expr/#op._S_expr) | Allows use of aggregation expressions within the query language. |
| [$jsonSchema](https://docs.mongodb.com/manual/reference/operator/query/jsonSchema/#op._S_jsonSchema) | Validate documents against the given JSON Schema. |
| [$mod](https://docs.mongodb.com/manual/reference/operator/query/mod/#op._S_mod) | Performs a modulo operation on the value of a field and selects documents with a specified result. |
| [$regex](https://docs.mongodb.com/manual/reference/operator/query/regex/#op._S_regex) | Selects documents where values match a specified regular expression. |
| [$text](https://docs.mongodb.com/manual/reference/operator/query/text/#op._S_text) | Performs text search. |
| [$where](https://docs.mongodb.com/manual/reference/operator/query/where/#op._S_where) | Matches documents that satisfy a JavaScript expression. |

{ "\_id" : 1, "category" : "food", "budget": 400, "spent": 450 }

{ "\_id" : 2, "category" : "drinks", "budget": 100, "spent": 150 }

{ "\_id" : 3, "category" : "clothes", "budget": 100, "spent": 50 }

{ "\_id" : 4, "category" : "misc", "budget": 500, "spent": 300 }

{ "\_id" : 5, "category" : "travel", "budget": 200, "spent": 650 }

{ $expr: { <expression> } }

db.monthlyBudget.find( { $expr: { $gt: [ "$spent" , "$budget" ] } } )

{ field: { $mod: [ divisor, remainder ] } }

db.inventory.find( { qty: { $ : [ 4, 0 ] } } )

{ <field>: { $regex: /pattern/, $options: '<options>' } }

{ <field>: { $regex: 'pattern', $options: '<options>' } }

{ <field>: { $regex: /pattern/<options> } }

# 排序

db.inventory.find().sort({item:1})

db.inventory.find().sort({qty:1, item:-1})

1是升序

-1是降序

# limit

读取指定数量

db.inventory.find().limit(3)

# skip

忽略指定数量

db.inventory.find().skip(1). limit(3)

# 聚合aggregate

Reference

Aggregation Pipeline Operators

模拟数据：

|  |
| --- |
| db.sales.insert([{ "\_id" : 1, "item" : "abc", "price" : 10, "fee" : 2, date: ISODate("2014-03-01T08:00:00Z") },{ "\_id" : 2, "item" : "jkl", "price" : 20, "fee" : 1, date: ISODate("2014-03-01T09:00:00Z") },{ "\_id" : 3, "item" : "xyz", "price" : 5, "fee" : 0, date: ISODate("2014-03-15T09:00:00Z") }]) |

add： 增加一个number

|  |
| --- |
| # 多个key相加  db.sales.aggregate(  [  { $project: { item: 1, total: { $add: [ "$price", "$fee" ] } } }  ]  )  # 固定数值相加  db.sales.aggregate(  [  { $project: { item: 1, total: { $add: [ "$price", 10 ] } } }  ]  ) |

abs：取绝对值

subtract：减法

|  |
| --- |
| # 测试数据  db.sales.insert([{start:5,end:8},{start:4,end:4},{start:9,end:7},{start:6,end:7}])  db.sales.aggregate([  {  $project: { delta: { $abs: { $subtract: [ "$start", "$end" ] } } }  }  ]) |

avg：求平均值

|  |
| --- |
| #测试数据  db.sales.insert([  {"item" : "abc", "price" : 10, "quantity" : 2, "date" : ISODate("2014-01-01T08:00:00Z") },  { "item" : "jkl", "price" : 20, "quantity" : 1, "date" : ISODate("2014-02-03T09:00:00Z") },  { "item" : "xyz", "price" : 5, "quantity" : 5, "date" : ISODate("2014-02-03T09:05:00Z") },  {"item" : "abc", "price" : 10, "quantity" : 10, "date" : ISODate("2014-02-15T08:00:00Z")},  {"item" : "xyz", "price" : 5, "quantity" : 10, "date" : ISODate("2014-02-15T09:12:00Z") }  ])  db.sales.aggregate(  [  {  $group:  {  \_id: "$item",  avgAmount: { $avg: { $multiply: [ "$price", "$quantity" ] } },  avgQuantity: { $avg: "$quantity" }  }  }  ]  ) |

# 创建集合

db.createCollection("new\_coll")

# 备份与恢复

## 备份：

mongodump -h dbhost -d dbname -o dbdirectory

-h：

MongDB所在服务器地址，例如：127.0.0.1，当然也可以指定端口号：127.0.0.1:27017

-d：

需要备份的数据库实例，例如：test

-o：

备份的数据存放位置，例如：c:\data\dump，当然该目录需要提前建立，在备份完成后，系统自动在dump目录下建立一个test目录，这个目录里面存放该数据库实例的备份数据。

mongodump -h 127.0.0.1 -d test -o d:\

mongodump -h 127.0.0.1:27017 -d test -o d:\

mongodump -h 127.0.0.1:27017 -d test –-collection counters -o d:\

## 恢复

mongorestore -h <hostname><:port> -d dbname <path>

--host <:port>, -h <:port>：

MongoDB所在服务器地址，默认为： localhost:27017

--db , -d ：

需要恢复的数据库实例，例如：test，当然这个名称也可以和备份时候的不一样，比如test2

--drop：

恢复的时候，先删除当前数据，然后恢复备份的数据。就是说，恢复后，备份后添加修改的数据都会被删除，慎用哦！

<path>：

mongorestore 最后的一个参数，设置备份数据所在位置，例如：c:\data\dump\test。

你不能同时指定 <path> 和 --dir 选项，--dir也可以设置备份目录。

--dir：

指定备份的目录

你不能同时指定 <path> 和 --dir 选项。